Homework 3

Natural Language Processing 2018/2019

Contacts

[bevilacqua | spadoni | scarlini | blloshmi] @di.uniroma1.it

If you write on the **Facebook group**

we will love you much more <3

What you will do

- Train your own multi-inventory Word Sense Disambiguation system!

Homework outline

- **Implement** and **train** a system capable of performing both **fine-grained** and **coarse-grained WSD.**
- **Evaluate** the model on both fine- and coarse-grained WSD using **MFS**
- Implement **predict** functions in the predict.py file to make predictions with your trained model for each inventory provided

takes in a sentence as input, output the sense for each word in that sentence.Two things - X; words. Y; the words, if it has senses, the words are replaced by the senses

- Write a **report** where you describe your approach and your result analysis
    - Text report **max 3 pages (4 for the big projects)**, text after the 3rd (4th) page will not be considered
    - Images and tables **max 3 pages**
- **Submit your code, trained model and report**
- **BIG PROJECT Only:** Deal with **multilingual** WSD

What we provide

- English sense-tagged **training corpora**
- A reference **paper**
- 5 English **evaluation datasets**
- Mapping from BabelNet synsets to WordNet 3.0 synsets (as in homework 2)
- Mappings from BabelNet synsets to coarse-grained inventories:
  - WordNet Domains
  - Lexicographer’s id
- These slides as guidelines

What we provide

For the big project only:

- 6 **multilingual evaluation datasets**
- Mapping from `lemmas` to BabelNet synsets for each other language (request it from us via email)

Word Sense Disambiguation

You have a sentence like:

I robbed a ***bank*** yesterday.

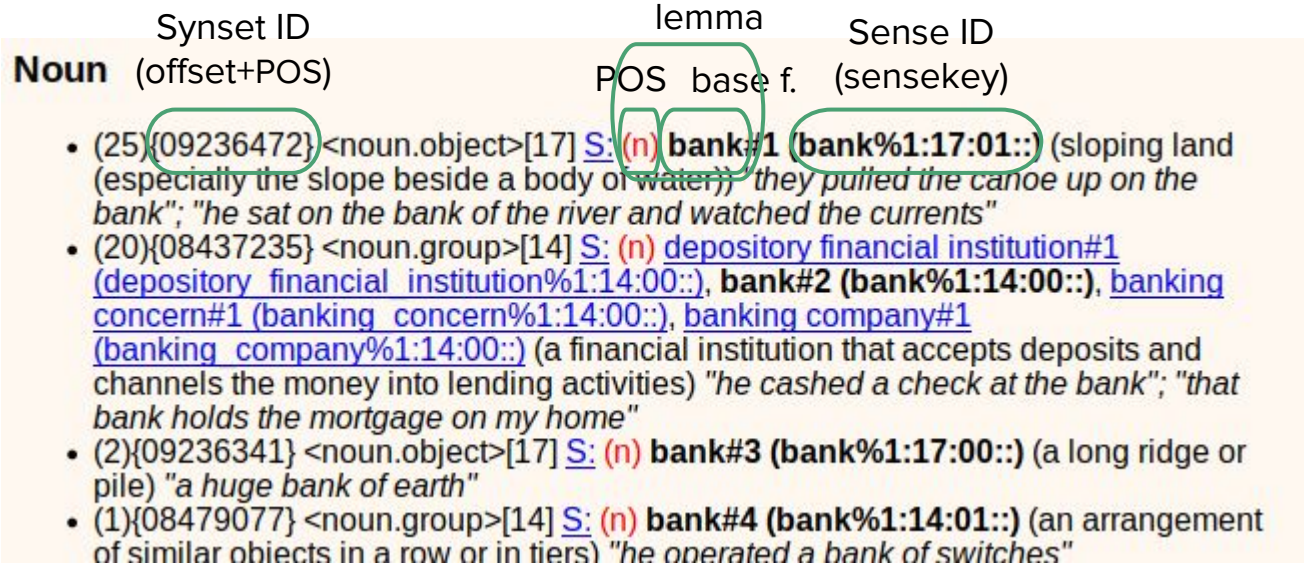
The noun ***bank*** is polysemous. It has more than one meaning.

Word Sense Disambiguation

- We would like a have system capable of automatically resolving ambiguity.
- We need to choose a sense inventory to pick senses for each word we want to disambiguate.
- For the homework we picked **WordNet**

WordNet

WordNet is a sense inventory. It maps lemmas, i.e. the couple of base form - POS underlying each surface form (like <bank, NOUN> for bank) to a set of possible senses.



Fine-grained vs Coarse-grained

- *Fine-grained* inventories contain lots of refined senses for every `lemma`, therefore they tend to suffer from over-specification and sparsity

- *Coarse-grained* inventories collect senses into more general categories thereby reducing the sparsity of the data. However, they usually end up with less specific (i.e. informative) senses for every `lemma`

Fine-grained WSD

Fine-grained all-words WSD data comes in the form of sentences where some words are associated with some of their senses.

```
<wf lemma="how" pos="ADV">How</wf>
<instance id="d000.s000.t000" lemma="long" pos="ADJ">long</instance>
<wf lemma="have" pos="VERB">has</wf>
<wf lemma="it" pos="PRON">it</wf>
<instance id="d000.s000.t001" lemma="be" pos="VERB">been</instance>
<wf lemma="since" pos="ADP">since</wf>
<wf lemma="you" pos="PRON">you</wf>
<instance id="d000.s000.t002" lemma="review" pos="VERB">reviewed</instance>
<wf lemma="the" pos="DET">the</wf>
<instance id="d000.s000.t003" lemma="objective" pos="NOUN">objectives</instance>
```

Fine-grained WSD

- Fine-grained all-words WSD data comes in the form of sentences where some words are associated with some of their senses.

```
d000.s000.t000                                                    long%3:00:02::
d000.s000.t001                                                      be%2:42:03::
d000.s000.t002                                                  review%2:31:00::
d000.s000.t003 objective%1:09:00::
```

- You can use the nltk package to get the WordNet synset from the sensekey:
  - `from nltk.corpus import wordnet as wn`
  - `synset = wn.synset_from_sense_key( "long%3:00:02::" )`
  - `synset_id = "wn:" + str(synset.offset()).zfill( 8) + synset.pos()`

    map synset_id to babelnetID's.

Fine-grained WSD

- The task requires to associate a word in context with one of the meanings enumerated in the sense inventory.
- The biggest difficulty is dealing with a input-dependent target classes -- this means sparsity.

Coarse-grained WSD

- Coarse-grained WSD tries to overcome the problems of traditional fine-grained WSD by providing general purpose labels, less strictly tied to lemmas.

- To exploit resources created for fine-grained WSD, coarse-grained inventories are generally structured as a mapping from WordNet synsets to coarse-grained labels.

Coarse-grained WSD

WordNet Domains:

http://wndomains.fbk.eu/

The labels are semantic domains (areas of human-knowledge), adapted from the Dewey Decimal Classification system, used for bibliographic purposes.

Coarse-grained WSD

WordNet Domains:

```
wn:01147060v sport                  #(parry)
wn:04840715n quality                #(grace)
wn:01701858v literature             #(rhyme)
wn:02456505n animals biology        #(genus_burmeisteria)
```

- N.B. Many synsets are mapped to the semantically empty `factotum` class

Coarse-grained WSD

Lexicographer's IDs (LexNames):

https://wordnet.princeton.edu/documentation/lexnames5wn

The labels are ontological classes (e.g. `noun.person` for the noun *scientist*)

Coarse-grained WSD

Lexicographer’s IDs (LexNames):

```
wn:01147060v verb.competition  #(parry)
wn:04840715n noun.attribute    #(grace)
wn:01701858v verb.creation     #(rhyme)
wn:02456505n noun.animal       #(genus_burmeisteria)
```

Data

We provide a comprehensive WSD Framework by Raganato et al., available at http://lcl.uniroma1.it/wsdeval/.

- The framework provides **Semcor**, a semantically annotated subset of the Brown Corpus, which can be used as **training** corpus.

- The framework also contains all the benchmark **Evaluation Datasets** for WSD (Senseval2, Senseval3, Semeval2007, Semeval2013 and Semeval2015).

Data

The Raganato et al. format:

The .xml file (needed for training and prediction)

```
<wf lemma="how" pos="ADV">How</wf>
<instance id="d000.s000.t000" lemma="long" pos="ADJ">long</instance>
<wf lemma="have" pos="VERB">has</wf>
<wf lemma="it" pos="PRON">it</wf>
<instance id="d000.s000.t001" lemma="be" pos="VERB">been</instance>
<wf lemma="since" pos="ADP">since</wf>
<wf lemma="you" pos="PRON">you</wf>
<instance id="d000.s000.t002" lemma="review" pos="VERB">reviewed</instance>
<wf lemma="the" pos="DET">the</wf>
<instance id="d000.s000.t003" lemma="objective" pos="NOUN">objectives</instance>
```

Data

The Raganato et al. format:

The .gold.key.txt file (needed for training only!)

```
d000.s000.t000 long%3:00:02::
d000.s000.t001 be%2:42:03::
d000.s000.t002 review%2:31:00::
d000.s000.t003 objective%1:09:00::
```

Data

You can use any other data for training:

- OMSTI (One Million Sense Tagged Instances): included in the Raganato et al. Framework
- Wordnet's Tagged Glosses: available as part of the UFSAC project
- T-O-M: huge but auT-O-Matically built
- EuroSense and SEW: also automatically built
- Anything else (be creative!) except for...

Evaluation data:

- Senseval-2 all-words, Senseval-3 all-words, SemEval 2007, SemEval 2013, SemEval 2015, which are to be used ONLY for **development (only one among them!)** and **testing.**

Data

Additional resources:

- The mapping from WordNet synsets to BabelNet ids (same as in homework 2).
- The mapping from BabelNet synsets to WordNet domains classes
- The mapping from BabelNet synsets to LexNames classes
- The mapping from `lemma` to BabelNet synsets
  - This mapping can be created by using *nltk.corpus.wordnet* (see later)
- The mapping from `lemma` to the MFS
  - Again, you can get the MFS with the *nltk.corpus.wordnet* package

The model

- You are free to choose whatever system you want to complete the homework.
- However, we choose a simple neural WSD system as baseline, which you can extend as you like.

Neural Sequence Learning Models for Word Sense Disambiguation

Raganato, Delli Bovi and Navigli, EMNLP 2017 **(link to the paper)**

The model

Other interesting papers you could draw inspiration from:

- Yuan et. al (2015), *Semi-supervised Word Sense Disambiguation with Neural Models*
- Melacci et al. (2018), *Enhancing Modern Supervised Word Sense Disambiguation Models by Semantic Lexical Resources*
- Luo et al. (2018), *Incorporating Glosses into Neural Word Sense Disambiguation*
- Vial et al. (2019), *Sense Vocabulary Compression through the Semantic Knowledge of WordNet for Neural Word Sense Disambiguation*

The model

- To improve the model you can resort to some pre-trained language models instead of standard word embeddings
- Some effective options are:
  - ELMo: word-based, straightforward to use
  - BERT: BPE-based, harder to set up
- The easiest way to integrate either of them is via the tensorflow_hub module
- They will replace your embedding layer

The model

Things we would love to see and would count as extra:

- Multitask learning! (there is much more than simply predicting everything at once, tensorflow is your limit!)
- Boost fine-grained with coarse-grained predictions!
- Any use of sense embeddings, especially those you trained yourself for homework 2!

N.B. all of the above are open research questions and we will love for you to answer them.

Evaluation

The evaluation in WSD is not as straightforward as you might think.

- Usually we have a softmax activation on the last layer, which yields a probability distribution, and we take the `argmax` of the entire output layer.
  - E.g. `car#noun`
    - `synset = argmax(prob)` `# prob` is the probability distribution over the entire output layer
- In WSD, we take the argmax of only the candidate synsets for the `lemma` we are trying to disambiguate
  - E.g. `car#noun` has `C=[bn:00007309n, bn:00015785n, bn:00015786n, bn:00015787n, bn:00014462n]`
    - `synset = argmax(prob[C])` `# prob` is the probability distribution over the entire output layer, but we restrict our attention only on the synsets in C and take the most likely one

Evaluation

- **Getting the list of candidate synsets for a given lemma POS:**

- **English**: you can exploit `nltk.corpus.wordnet`, retrieve the associated synsets' offsets and pos, and map them to BabelNet ids through the mapping we provided.
- **Multilingual**: you will have to contact us to have access to the mapping from a multilingual lemma POS to BabelNet ids.

MFS

- Whenever you come across some `lemma`s you have never seen during training, you cannot give an answer to those instances
- In order to provide an answer you need a **backoff** strategy
- The most common option as a backoff strategy is the **M**ost **F**requent **S**ense
  - The MFS always returns the predominant sense of a `lemma`
- Therefore, whenever you encounter a `lemma` that was not in the training, you give its MFS as output
- By doing so, you provide an answer for EVERY instance

Metric

- The performances on the Evaluation Datasets are reported as **F1** scores
- As a quick recap F1 is the harmonic mean between precision (**P**) and recall (**R**)
  - Precision is the number of correct answers over the number of answers your system *gives*
    - P = TP / (TP+FP)
  - Recall is the number of correct answers over the total correct answers your system *should give*
    - R = TP / (TP+FN)
  - Then
    - F1 = 2*P*R/(P+R)
- Since you **MUST** implement the MFS backoff strategy, you will have *P = R = F1*
  - (because your system always provides an answer!)

Metric

- To check the performance of the model, you can use the official Java scorer of the Raganato et al. Framework, which you find in the release along with the instructions to use it.

- **N.B: both predictions and gold have to be in the same format we will describe in a few slides.**

Predict

- You will have to provide an implementation for the `predict_babelnet`, `predict_wordnet_domains`, `predict_lexicographer` functions in `predict.py`. Each shares the same signature:

```
def predict(input_path:str, output_path:str, resources_path:str) -> None:
```

Predict

Predict functions parameters:

- `input_path`: will be the path of our secret test set, which will be an .xml in the Raganato et al. format
- `output_path`: the path where you will write your predictions
- `resources_path`: we will pass the path to your resources directory.

Predict

The output format is the same as the format of the .gold.key.txt files in the Raganato et al. framework BUT THE PREDICTIONS WILL NOT BE SENSEKEYS!.

N.B. For each instance in the input file you will need to produce one (and only one) prediction. Any time you do not have an output, il will be counted as wrong.

- `predict_babelnet`: the predictions will be babelnet ids (remember you have a mapping to handle that!)
- `predict_wordnet_domains`: the predictions will be WordNet Domains
- `predict_lexicographer`: the predictions will be LexNames labels.

Predict

The prediction format for **babelnet** inventory must be as follow:

`predict_babelnet` output example:

```
d002.s013.t003 bn:00987654n
d002.s013.t004 bn:00112233n
d002.s013.t005 bn:00987654n
d003.s013.t001 bn:01234567r
```

predict_babelnet, predict_wordnet_domains, predict_lexicographer

Predict

The prediction format for **wordnet domains** inventory must be as follow:

`predict_wordnet_domains` output example:

```
d002.s013.t003 animal
d002.s013.t004 sport
d002.s013.t005 biology
d003.s013.t001 factotum
```

Predict

The prediction format for **lexicographer** inventory must be as follow:

`predict_lexicographer` output example:

```
d002.s013.t003 noun.person
d002.s013.t004 noun.animal
d002.s013.t005 noun.group
d003.s013.t001 adv.all
```

Big Project (non-attending students only)

- You will handle *multilingual* data for Italian, French, German and Spanish
- The use of *sense embeddings* is now mandatory, not an extra
- You must implement *multitask learning*, not an extra
- For the multilingual setting you will have to implement the function `predict_multilingual` as well:

```
def predict_multilingual(
    input_path:str,
    output_path:str,
    resources_path:str, lang:str) -> None:
```

Big Project (non-attending students only)

- The lang parameter will be one of ‘it’, ‘de’, ‘es’ and ‘fr’, depending on the language of the evaluation corpus.

- The predictions need be in the same format as predict_babelnet

```
d002.s013.t003 bn:00987654n
d002.s013.t004 bn:00112233n
d002.s013.t005 bn:00987654n
d003.s013.t001 bn:01234567r
```

- You will probably need to use the mapping from <lemma, POS> to BabelNet ids, which you have to request by contacting us.

What we expect in the report

- **Describe** any important step you carried out in the **preprocessing** phase
- Report the **model** and the parameters you used, plus any implementation choice worth mentioning (e.g. design choices, additional features, use of sense embeddings, ecc.)
- Report considerations on the difference in performances and results that you get from the various **sense inventories**
- Report some **interesting plots** such as the tables, charts and anything that adds value to the work you have done

Submission

As in Homework 1, we provide a skeleton for your submission which YOU HAVE TO FOLLOW. If you don’t, you will be penalized.

The project will have this **structure**:

- Your report in **pdf** format, named **report.pdf**
- A folder with your source code named **code**, it also has to include the scripts we provided
- A folder containing the resources necessary to run your model and make predictions with the *predict.py* script, named **resources**

Submission

- Register to GitLab (you should have already done that :D)
- Create a new repository (**private**) with name: <**firstname**>_<**lastname**>_<**matricola**>_**nlp19hw3**
- **Share** the project with us (*MAINTAINER role)*:
  - use ALL the emails on the second slide and
  - navigli@di.uniroma1.it
- The link where you have to submit:

SUBMISSION FORM

How we will grade

The maximum grade for this homework is 34.5 (115% of 30) weighted as follows:

- Quality, comments and cleanness of code [30%]
- Report [55%]
- Overall performance of the system [15%]
    - We will evaluate you on a secret WSD test set
- **Creativity boost:** Improvements over the model [15%]
    - You will get extra points if you add some effective features to your model, in this case we will expect a comparison with the different approaches and meaningful observations on what you experimented

Deadlines

- We will upload everything you need to complete this homework Sunday 2 June evening the latest, on the facebook group
- Please check http://naviglinlp.blogspot.com/p/natural-language-processing-basic.html for submission and presentation dates
- For those doing the Homework
  - There are 3 possible submission dates
  - 1 week after your submission you will have to present your Homework with a presentation
- For those doing the project
  - There are 5 possible submission dates
  - 1 week after your submission you will have to present your project with a presentation

Presentation

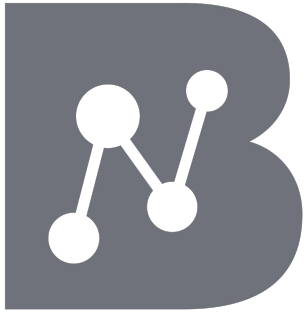
- Presentation will be done in classroom with slides
- Each student will have a maximum of 10 minutes for presentation (strict) +10 minutes for questions from the Professor
- You should present a summary of your work on the slides
    - Make a presentation out of your report

Final Grade

- Your Homework 3 will be graded by us and after the presentation Prof. Navigli will set your final grade
- If you did all the Homeworks your final grade will be the average of the three
- For those doing the big project: this will be your only final grade, determined by the submission and the presentation

Motivation

- A (super!) fancy **BabelNet T-Shirt** for all the students with final grade > 30 !



We will check all your submissions for **plagiarism**!

- If we find that you plagiarised you are **OUT** of this year's course and you cannot take the exam, you will have to sign up for the course next year
- We have a **zero-tolerance** policy for plagiarism!

- **Advice**: try to make many experiments to get the most out of this course
- **Setting up** the model might be tricky and bugs are always around the corner, so make sure you double check every step of your pipeline
- **Memory** might also be an issue: restrict your output vocabulary as much as you can (e.g. only add to the output vocab the synsets that are in the training set)

Good luck!

If you have any questions, do not hesitate to post them on the Facebook group